As an input for assignment I used the given kmeans.csv file having 5 groups a, b, c, d, e - Figure 1-

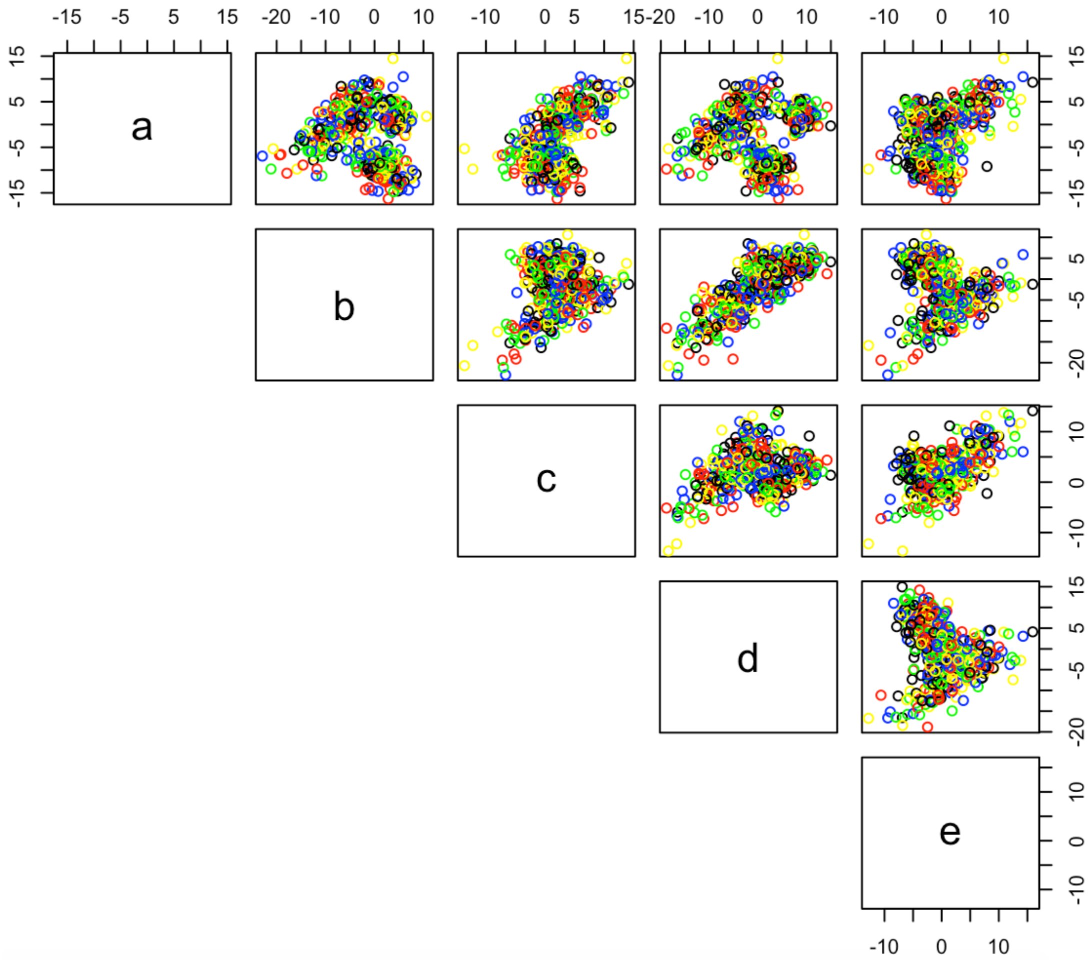

Figure 1 representing 5 groups using pairs

I used 20 iterations and 3 clusters in order to calculate which point belongs to the cluster with the nearest mean.
as a first step I used my.kmeans function, In order to assign each point to the cluster with the least squared distance between the mean and the point.

```
my.kmeans <- function(X, k) {
#take the first mean of a sample of the given points
  means <- X[sample(nrow(X), k),]
#Initialize an old mean to the current to keep track
of the mean values of the samples
  old_means <- means
#Initialize the clusters vector
  cluster <- NULL
#start 20 iterations
  while (iterations > 0) {
#looping over observations k times
    for (j in k) {
      for (i in 1:nrow(X))
      {
#initialize minimum distance to large number
        min_dist = 10e9
#looping over means
```

```
for (m in 1:nrow(means))
{#calculate distance from the point to the mean
       d_from_mean = sum((means[m,] - X[i,]) ^ 2)
#check if the calculated mean is the closest mean to th
e point
        if (d_from_mean <= min_dist){
#assign this mean to the point "nearest mean"
            cluster[i] = m
#change the minimum distance
            min_dist = d_from_mean
          }}}}
#the points coordinates are the mean of the calculated
in the cluster
    means <-t(sapply(1:k, function(c)
   apply(X[cluster == c,], 2, mean)))
#update the  iterator index, and old means
    iterations <- iterations - 1
    old_means <- means
  }return(list(means = means, cluster = cluster))
}
```

And this step in order to calculate nearest mean vector for the 3 clusters on the given points with an output.

	a	b	c	d	e
[1,]	-1.012748	-8.626076	1.0630611	-7.617423	0.80054348
[2,]	2.986158	2.487422	4.6498060	6.027560	-0.09022683
[3,]	-9.420448	1.672011	0.5089758	1.919512	-1.23251920

After that in each iteration the following steps was followed:

- Assign each point to closest center.
- initialize the covariance matrix using the initialized points clusters.
- calculate the first iteration covariance matrix the summation E((x − meanx)(y − meany)).
- calculate the cluster probability from the mean and cov matrix.
- use the calculated clusters for reoptimizing the covariance and mean metrices.
- calculate the covariance matrix the summation E((x − meanx)(y − meany)) again.
- iterate until converges.

Other functions were used to help in the calculations:

- calculate covariance matrix for the multivariate distribution
- multivariate distribution probability density function

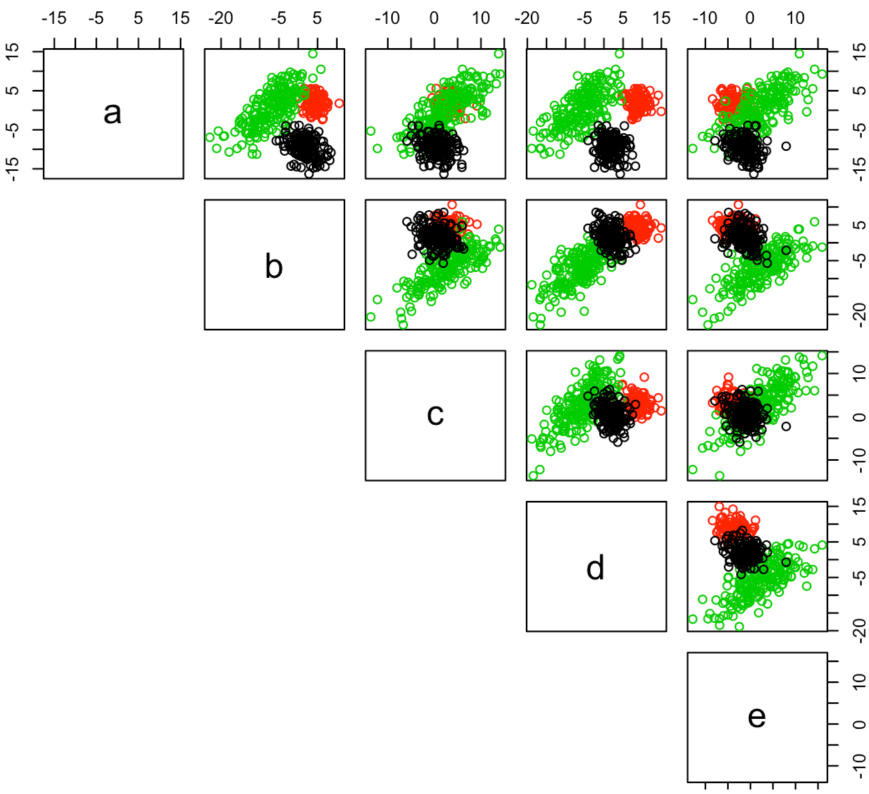

Figure 2 Clusters after 20 iterations

with the three clusters covariance matrix as an output.